In the claims:

For the Examiner's convenience, all pending claims are presented below with changes shown.

1. (Currently Amended) A method comprising:

inserting a single instruction at a start of a block of code~~, performing only one test for the block of code~~ to determine if resources of a processor ~~an architectural stack~~ are available for the block of code, wherein the block of code ~~including~~ includes multiple instructions ~~adding data to the stack or removing data from the stack~~; and

if the resources are available, modifying the available resources according to requirements of the multiple instructions in the block of code ~~signaling an error if said resources of the architectural stack needed for said block of code are not available~~.

2. (Currently Amended) The method ~~as claimed in~~of claim 1, ~~said method~~ further comprising:

determining a set of available resources that will be available after said block of code has executed.

3. (Canceled)

4. (Currently Amended) The ~~method as claimed in~~of claim 1 wherein the availability of the ~~stack~~ processor resources are determined at ~~a~~ compile time.

5. (Currently Amended) The method ~~as claimed in~~of claim 1 wherein the availability of the ~~stack~~ processor resources are determined dynamically.

6. (Currently Amended) The method ~~as claimed in~~of claim 1 further comprising: ~~wherein~~ signaling ~~said~~ an error message if ~~said~~ the resources of the ~~architectural stack~~ processor needed for ~~said~~ the block of code are not available; and ~~comprises~~

in response to the error message, branching to a fault handler routine.

7. (Currently Amended) The method ~~as claimed in~~of claim 6 wherein signaling of said fault handler routine simulates a processor exception.

8. (Currently Amended) The method ~~as claimed in~~of claim 1 wherein the ~~stack~~ resources are represented by a bit vector.

9. (Currently Amended) The method ~~as claimed in~~ of claim 8 wherein said bit vector is generated dynamically.

10. (Currently Amended) A computer-readable medium having stored thereon a set of instructions to monitor processor resources, said set of instruction, which when executed by a processor, cause said processor to perform a method comprising:

inserting a single instruction at a start of a block of code~~, performing only one test for the block of code~~ to determine if resources of ~~an architectural stack~~the processor are

available for the block of code, wherein the block of code ~~including~~ includes multiple instructions ~~adding data to the stack or removing data from the stack~~; and

if the resources are available, modifying the available resources according to requirements of the multiple instructions in the bock of code~~signaling an error if said resources of the architectural stack needed for said block of code are not available~~.

11. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 10, wherein said set of instructions further includes additional instructions, which when executed by said processor, cause said processor to perform said method further comprising:

determining a set of available resources that will be available after said block of code has executed.

12. (Canceled)

13. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 10 wherein the availability of the ~~stack~~ processor resources are determined at ~~a~~ compile time.

14. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 10 wherein the availability of the ~~stack~~ processor resources are determined dynamically.

15. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 10 wherein additional instructions, which when executed by the processor, cause the processor to perform the method further comprising:

signaling ~~said~~ an error message if ~~said~~ the resources of the ~~architectural stack~~processor needed for ~~said~~ the block of code are not available; and ~~comprises~~

in response to the error message, branching to a fault handler routine.

16. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 15 wherein signaling of said fault handler routine simulates a processor exception.

17. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 10 wherein the ~~stack~~ resources are represented by a bit vector.

18. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 17 wherein said bit vector is generated dynamically.

19. (Currently Amended) A computer-readable medium, having stored thereon a first set of instructions, the first set of instructions, which when executed by a processor, generate a second set of instructions through a binary translation process, the second set of instructions when executed by the processor, cause said processor to perform a method comprising:

inserting a single instruction at a start of a block of code~~, performing only one test for the block of code~~ to determine if resources of ~~an architectural stack~~ the processor are available for the block of code, wherein the block of code ~~including~~ includes multiple instructions ~~adding data to the stack or removing data from the stack~~; and

if the resources are available, modifying the available resources according to requirements of the multiple instructions in the bock of code~~signaling an error if said resources of the architectural stack needed for said block of code are not available~~.

20. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 19, wherein said set of instructions further includes additional instructions, which when executed by said processor, cause said processor to perform said method further comprising:

determining a set of available resources that will be available after said block of code has executed.

21. (Canceled)

22. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 19 wherein the availability of the ~~stack~~ processor resources are determined dynamically.

23. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 19 wherein additional instructions, which when executed by the processor, cause the processor to perform the method further comprising:

signaling ~~said~~ an error message if ~~said~~ the resources of the ~~architectural stack~~processor needed for ~~said~~ the block of code are not available; and ~~comprises~~

in response to the error message, branching to a fault handler routine.

24. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 23 wherein signaling of said fault handler routine simulates a processor exception.

25. (Currently Amended) The computer-readable medium ~~as claimed in~~of claim 19 wherein needed processor resources are represented by a bit vector.